MPRI - Probabilistic Programming Language
Adapted from 2024 final exam

Instructions There are 2 independent exercises: one on modeling and one on semantics. The exam is long and you are not expected to answer every question. Questions are numbered 1 to 27. In the margin you find scales and levels of the different part.

– Electronic devices must be turned off and put in your bag.

– Authorized documentation: course slides.

– Your answers can be in French or in English as you wish.

– The questions are numbered: report these numbers on your copy.

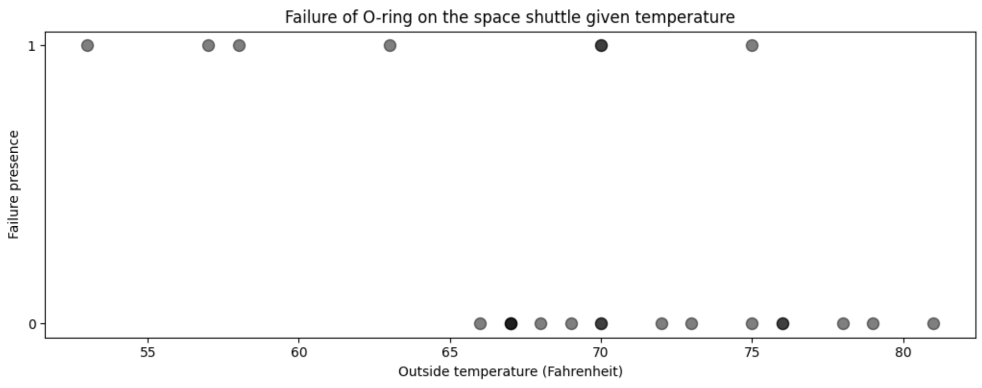
– The scale is given on an indicative basis and can be modified.

I). Programming Bayesian models

[4 pt] Course Application

A. Challenger Space Shuttle Disaster

On January 28, 1986, the space shuttle Challenger exploded shortly after liftoff. The explosion was caused by the failure of an O-ring in a field joint which was sensitive to the outside temperature. Data on failures of O-rings was available from the previous twenty-four flights.
They are represented on the following figure:



We propose to use a logistic regression and to model the data with a sigmoid function of the temperature:

$$p(t) = \frac{1}{1 + e^{bt+a}}$$

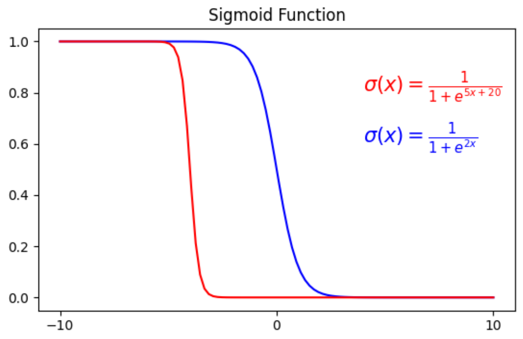
a and b are prior parameters that will follow a Gaussian distribution centered in 0 and with variance 30.

```
temps = [66, 70, 69, 68, 67, 72, 73, 70, 57, 63, 70,
  78, 67, 53, 67, 75, 70, 81, 76, 79, 75, 76, 58];
fails = [false, true, false, false, false, false, false, false, true, true, true,
  false, false, true, false, false, false, false, false, false, true, false, true];

def model(temps_obs, fails_obs):
   ...

with ... :
   dist = infer(model, temps, fails)
```

FIGURE 1 – Inference of the probability of crash for the Challenger given the data



1. Complete the . . . in the program presented in Figure 1 written in μPPL to answer the question:

   What is the probability that there is a failure at the liftoff temperature of 31^o F ?

II). Soundness of Inference

In the following we compute the distributions described by probabilistic programs using different semantics describing the mathematical semantics (distribution semantics) or the result of the inference based on the sample trace or on enumeration. The purpose of this exercise is to show (first in the simple discrete case) that the inference algorithms are sound with respect to the mathematical semantics. This approach does not say anything on the efficiency of these algorithms.

Core calculus

Consider the syntax of the following probabilistic lambda-calculus

$$
\begin{aligned}
e := {} & x \mid \lambda x\, e \mid (e)\, e \mid \texttt{fix}\, e \;\mid 0 \mid \texttt{succ}\; M \mid p\ (\forall p \in \mathbb{R}^+) \\
& \texttt{true} \mid \texttt{false} \mid \texttt{if}\; e\; \texttt{then}\; e\; \texttt{else}\; e \\
& \ell \mid \texttt{match}\; e\; \texttt{with}\; \{\ell\; x \to e \mid \cdots\} \\
\ell := {} & \texttt{nil} \mid \texttt{cons}\; e\; e \mid \texttt{Leaf}\; e \mid \texttt{Node}(e, e)
\end{aligned}
$$

```
let x = sample flip in
 let y = sample flip in
  let z = sample flip in
   let r = 1 + z + 2 * y + 4 * x in
    if r ⩽ 6
    then let _ = score(0.15) in Dirac r
    else let _ = score(0.05) in Dirac r
```

(a) Dice

```
let x = sample flip in
  if x then flip
  else let z = sample flip in
   if z
   then let _ = score 1.3 in Dirac z
   else let _ = score 0.7 in Dirac z
```

(b) Coin

FIGURE 2 – Examples of Discrete Probabilistic Programs

We will use standard operations on lists such as concatenation of lists, fold, ... together with the following type system

$$t := 1 \mid \texttt{int} \mid \texttt{bool} \mid \texttt{List}\, t \mid \texttt{Tree}\, t \mid t \to t \mid t \times t$$

$$\textsc{Var}\ \frac{}{G, x : t \vdash x : t} \qquad \textsc{App}\ \frac{G \vdash e_1 : t_1 \qquad G \vdash e_2 : t_1 \to t_2}{G \vdash e_2(e_1) : t_2} \qquad \textsc{Abs}\ \frac{G, x : t_1 \vdash e : t_2}{G \vdash \lambda x.e : t_1 \to t_2}$$

$$\textsc{Fix}\ \frac{G \vdash e : t \to t}{G \vdash \texttt{fix}\, e : t} \qquad \textsc{Unit}\ \frac{}{G \vdash () : 1} \qquad \textsc{Pair}\ \frac{G \vdash e_1 : t_1 \qquad G \vdash e_2 : t_2}{G \vdash (e_1, e_2) : t_1 \times t_2}$$

$$\textsc{True}\ \frac{}{G \vdash \texttt{true} : \texttt{bool}} \qquad \textsc{False}\ \frac{}{G \vdash \texttt{false} : \texttt{bool}} \qquad \textsc{If}\ \frac{G \vdash e_0 : \texttt{bool} \qquad G \vdash e_1 : t \qquad G \vdash e_2 : t}{G \vdash \texttt{if}\ \texttt{e}_0\ \texttt{then}\ \texttt{e}_1\ \texttt{else}\ \texttt{e}_2 : t}$$

We give the standard set semantics to the types that are interpreted as sets:

$$[\![\texttt{int}]\!] = \mathbb{N} \qquad [\![\texttt{t}_1 \to \texttt{t}_2]\!] = \text{functions from } t_1 \text{ to } t_2 \ldots$$

A. Discrete bayesian inference

[8 pt] Intermediate

We use the monadic approach to give a semantics to probabilistic programs. The let binding stands for first sample e_1, then use the output sample in e_2. The return statement that embeds base types t into probabilistic types $T\,t$ is defined through $\texttt{Dirac}\ e$ which stands for the random variable that evaluates e into a value v and outputs the value of v with probability 1.

$$\textsc{Sample}\ \frac{G \vdash e_1 : T\,t_1 \qquad G, x : t_1 \vdash e_2 : T\,t_2}{G \vdash \texttt{let x = sample e}_1\ \texttt{in e}_2 : T\,t_2} \qquad \textsc{Dirac}\ \frac{G \vdash e : t}{G \vdash \texttt{Dirac} : T\,t}$$

In this section, we will consider discrete probability type $T\,t$ and extend the core calculus with discrete probability constructions: a fair coin `flip` and a soft conditioning `score`.

$$\textsc{Fair Coin}\ \frac{}{G \vdash \texttt{flip} : T\,\texttt{bool}} \qquad \textsc{Conditioning}\ \frac{G \vdash p : \mathbb{R}^+}{G \vdash \texttt{score}\, p : T\,\texttt{unit}}$$

2. What distribution is represented by the `Dice` and `Coin` programs presented in Figure 2 ? (You don't have to justify your answer).

A.1. Mass semantics

We consider the standard mass semantics as the reference semantics. In the mass semantics, the closed programs of type Tt are interpreted in $\text{Mass}(t)$ as valuations with finite support, and represent the un-normalized score of each possible results in X.[1] We denote $\texttt{supp}(\mu) = \{v \in [\![t]\!] \mid \mu(v) \neq 0\}$.

$$\text{Mass}(t) = \{\mu : [\![t]\!] \to \mathbb{R}^+ \mid \texttt{supp}(\mu) \text{ is finite. }\}$$

We recall the semantics of terms: $\forall \gamma \in [\![G]\!]$

$$\begin{aligned}
[\![G \vdash \texttt{flip}]\!]^{\text{M}}_\gamma &= \begin{cases} \texttt{true} & \mapsto \frac{1}{2} \\ \texttt{false} & \mapsto \frac{1}{2} \end{cases} \\
[\![G \vdash \texttt{score}\, p]\!]^{\text{M}}_\gamma &= () \mapsto [\![p]\!]_\gamma \\
[\![G \vdash \texttt{Dirac}\, e]\!]^{\text{M}}_\gamma &= \begin{cases} \text{v where } [\![e]\!]_\gamma = \{\text{v}\} & \mapsto 1 \\ \_ & \mapsto 0 \text{ otherwise} \end{cases} \\
[\![G \vdash \texttt{let}\, x = \texttt{sample}\, e_1\, \texttt{in}\, e_2]\!]^{\text{M}}_\gamma &= \sum_{a \in \texttt{supp}([\![e_1]\!]_\gamma)} [\![e_1]\!]_\gamma\,(a)\, [\![e_2]\!]_{\gamma, x \mapsto a}
\end{aligned}$$

3. Compute the Mass semantics of the `Dice` and the `Coin` programs defined in Figure 2.
   (Give the main steps of the computations).

This semantics is sound with respect to the operational semantics presented in the lectures. It is an accurate semantics but realistic implementations cannot compute mass functions for infinite data types. The next two semantics manipulate more realistic representations of the programs. The purpose of this first part is to prove that they are sound by reduction to the mass semantics.

A.2. Trace semantics

Consider the sample trace semantics where closed programs of type Tt are interpreted in $\text{Trace}(t)$ as finite trees whose leaves are labeled with weighted values of type t.

$$\text{Trace}(t) = \texttt{Tree}([\![t]\!] \times \mathbb{R}^+) = \left\{\texttt{Leaf}([\![t]\!] \times \mathbb{R}^+) \mid \texttt{Node}(\text{Trace}(t) \times \text{Trace}(t))\right\}$$

Each node represents a probabilistic choice made when sampling `flip` and each leave contains the final value and the total weight of the trace of samples for the branch.
The let binding is a grafting operation which substitutes each leaf of the tree $[\![e]\!]^{\text{T}}_\gamma \in \text{Trace}(t)$ with the term $[\![e']\!]^{\text{T}}_{\gamma, x \leftarrow a}$ according to the value of the leaf; after re scaling it according to the weight of the leaf.

4. Semantics is defined by induction. Give the interpretation of the following terms:
   – $[\![G \vdash \texttt{flip}]\!]^{\text{T}}_\gamma \in \text{Trace}(\texttt{bool})$ – $[\![G \vdash \texttt{Dirac}\, e]\!]^{\text{T}}_\gamma \in \text{Trace}(t)$
   – $[\![G \vdash \texttt{score}\, p]\!]^{\text{T}}_\gamma \in \text{Trace}(\texttt{1})$ – $[\![G \vdash \texttt{let}\, x = \texttt{sample}\, e\, \texttt{in}\, e']\!]^{\text{T}}_\gamma \in \text{Trace}(t')$
   (You can draw trees to answer these questions and define auxiliary recursive functions).
5. Compute the sample trace semantics of the programs defined in Figure 2.
   (Give enough intermediate steps in the computations to show that you are following the write semantics).

In order to prove the correctness of the trace semantics, we will refer to the mass semantics and reason by induction on terms: it is sufficient to introduce a function $m^T : \text{Trace}(t) \to \text{Mass}(t)$ which is compatible with the probabilistic constructions (sampling, dirac, flip and score) to prove that the trace semantics is sound.

1. The inference would re-normalize the valuation to get a distribution. We work up to re-normalization for this section.

6. Define such a function $m^T : \mathrm{Trace}(t) \to \mathrm{Mass}(t)$ and explain its action.
   (You may use examples as guidelines.)
7. Prove the following cases:
   – $m^T([\![\texttt{Dirac}\, e]\!]^{\mathrm{T}}) = [\![\texttt{Dirac}\, e]\!]^{\mathrm{M}}$
   – $m^T([\![\texttt{flip}]\!]^{\mathrm{T}}) = [\![\texttt{flip}]\!]^{\mathrm{M}}$
   – $m^T([\![\texttt{score}\, p]\!]^{\mathrm{T}}) = [\![\texttt{score}\, p]\!]^{\mathrm{M}}$
   – $m^T([\![\texttt{let}\, x \,=\, \texttt{sample}\, e\, \texttt{in}\, e')]\!]^{\mathrm{T}}) = [\![\texttt{let}\, x \,=\, \texttt{sample}\, e\, \texttt{in}\, e']\!]^{\mathrm{M}}$

   (For the let binding case, you may first establish relations between the set of values in the leaf and the support, and the weights of values in the tree and domain of the valuation. You may then use the induction hypothesis on e and then on e'.)

A.3. Enumeration semantics

Consider the enumeration semantics where closed programs of type $T t$ are interpreted in $\mathrm{Enum}(t)$ as finite lists whose elements are labeled with weighted values of type t.

$$\mathrm{Enum}(t) = \texttt{List}([\![t]\!] \times \mathbb{R}^+) = \left\{\texttt{nil} \mid (([\![\mathtt{t}]\!] \times \mathbb{R}^+) \times \mathrm{Enum}(\mathtt{t}))\right\}$$

Each element contains a final value and a weight for this value. Values might be repeated.

8. Semantics is defined by induction. Give the interpretation of the following terms:
   – $[\![G \vdash \texttt{flip}]\!]^{\mathrm{E}}_\gamma \in \mathrm{Enum}(\texttt{bool})$
   – $[\![G \vdash \texttt{score}\, p]\!]^{\mathrm{E}}_\gamma \in \mathrm{Enum}(\texttt{1})$
   – $[\![G \vdash \texttt{Dirac}\, e]\!]^{\mathrm{E}}_\gamma \in \mathrm{Enum}(t)$
   – $[\![G \vdash \texttt{let}\, x = \texttt{sample}\ e\, \texttt{in}\, e']\!]^{\mathrm{E}}_\gamma \in \mathrm{Enum}(t')$

   For the case of the let binding, give an explanation of your definition.
9. Compute the Enumeration semantics of the programs defined in Figure 2.
   (You should give enough computation steps to illustrate your definitions.)
10. Define a function $m^E : \mathrm{Enum}(t) \to \mathrm{Mass}(t)$ that preserves the semantics.
11. The prove that semantics is preserved is by indudction. Prove the following cases:
   – $m^E([\![\texttt{Dirac}\, e]\!]^{\mathrm{E}}) = [\![\texttt{Dirac}\, e]\!]^{\mathrm{M}}$
   – $m^E([\![\texttt{let}\, x \,=\, \texttt{sample}\, e\, \texttt{in}\, e')]\!]^{\mathrm{E}}) = [\![\texttt{let}\, x \,=\, \texttt{sample}\, e\, \texttt{in}\, e']\!]^{\mathrm{M}}$
   – $m^E([\![\texttt{flip}]\!]^{\mathrm{E}}) = [\![\texttt{flip}]\!]^{\mathrm{M}}$
   – $m^E([\![\texttt{score}\, p]\!]^{\mathrm{E}}) = [\![\texttt{score}\, p]\!]^{\mathrm{M}}$

B. Continuous Bayesian Inference

[5 pt] Interme-diate

For the continuous case, we replace the fair coin `flip` by a uniform distribution over $[0, 1]$ `unif`:

$$\frac{\textsc{Uniform}}{\overline{G \vdash \texttt{unif} : T\,\mathbb{R}}}$$

12. What distribution is represented by the `Lreg` program presented in Figure 3 ? (Explain your answer.)

```
let s = sample unif in
    let b = sample unif in
        let _ = score(1 / (1 + exp(-abs(s * x + b - y))) in
              Dirac (s, b)
```

FIGURE 3 – `Lreg`, an example of continous probabilistic program.

B.1. Quasi-Borel Spaces, a semantics for continous probability

We will use in this section a conservative extension of measurable spaces and functions which is compatible with higher-order.
A quasi-borel space (QBS) is a set X together with a set M_X of functions from $\mathbb{R}$ to X so-called random elements such that

(A) all the constant functions are in M_X:

$$\forall x \in X,\ \lambda r^{\mathbb{R}}.x \in M_X$$

(B) M_X is closed by precomposition with measurable functions: for all measurable $f : \mathbb{R} \to \mathbb{R}$,

$$\forall \alpha \in M_X,\ \alpha \circ f \in M_X$$

(C) M_X satisfies the piecewise condition:
if $\uplus U_i$ is a countable partition of $\mathbb{R}$ where for all i, U_i is measurable and $\alpha_i \in M_X$, then the piecewise function β defined by if $r \in U_i$, then $\beta(r) = \alpha_i(r)$ satisfies $\beta \in M_X$

A QBS morphism $f : X \to Y$ is a function that respects the structure:

$$\forall \alpha \in M_X,\ f \circ \alpha \in M_Y$$

It is easy to check that **QBS** is a category. We denote as $\mathbf{QBS}(X, Y)$ QBS morphisms from X to Y.

Base types. The interpretation of Base types is: [2]

$$[\![\texttt{int}]\!]^{D} = [\mathbb{N}, M_{\mathbb{N}} = \{\alpha : \mathbb{R} \to \texttt{int} \mid \alpha \text{ is measurable }\}]$$
$$[\![\mathrm{R}]\!]^{D} = [\mathbb{R}, M_{\mathbb{R}} = \{f : \mathbb{R} \to \mathbb{R} \mid f \text{ is measurable }\}]$$

13. Describe the set of morphisms $\mathbf{QBS}(\mathbb{R}, \mathbb{R})$ from $\mathbb{R}$ to $\mathbb{R}$ endowed with its QBS structure.

Higher-order. Let us recall that the category **SET** is cartesian closed when endowed with the usual cartesian product, the function space is the set of functions and curryfication, uncurryfication and evaluation are standard.
The cartesian product of X and Y is defined as: the set $X \times Y$, the projections $\pi_X : X \times Y \to X$ and $\pi_Y : X \times Y \to Y$ together with the random elements

$$M_{X \times Y} = \{\alpha : \mathbb{R} \to X \times Y \mid \pi_X \circ \alpha \in M_X,\ \pi_Y \circ \alpha \in M_Y\}$$

Endowed with this structure, we can remark that projections are QBS morphisms.
Let $\mathrm{uncurry}()$ be the set uncurryfication.
The function space $[X, Y]$ is the set of QBS morphisms and

$$M_{[X,Y]} = \{\alpha : \mathbb{R} \to [X, Y] \mid \mathrm{uncurry}(\alpha) : \mathbb{R} \times X \to Y \text{is a QBS morphism}\}$$

2. A function $\alpha : \mathbb{R} \to \texttt{int}$ is measurable when $\forall n,\ \alpha^{-1}(n)$ is measurable in $\mathbb{R}\}$

14. Prove that the function space satisfies the first two conditions of QBSes.[3]
15. Let curry be the set curryfication and ev be the set evaluation. Prove that:

$$\forall f \in \mathbf{QBS}(X \times Y, Z),\ \mathrm{curry}(f) \in \mathbf{QBS}(X, [Y, Z])$$

16. To prove that $\mathrm{ev} \in \mathbf{QBS}(Y \times [Y, Z], Z)$ use that

$$\forall \alpha \in M_{Y \times [Y,Z]},\ \mathrm{ev} \circ \alpha = \mathrm{uncurry}(\pi_{[Y,Z]} \circ \alpha) \circ (\mathrm{id}_{\mathbb{R}} \times (\pi_Y \circ \alpha))$$

(That you may admit).

By consequent, **QBS** is a cartesian closed category that is suitable to interpret higher-order. Actually, the semantics of deterministic programs in **QBS** is equal to the usual **SET** semantics.

B.2. Measure semantics

[3pt] To the end Advanced

We denote by $\mathrm{Dist}(\mathbb{R})$ the measurable space of all S-finite measures on the measurable space $\mathbb{R}$ of all real numbers.
We consider the measure semantics reformulated in the **QBS** setting as the reference semantics. A QBS measure is an S-finite measure μ on $\mathbb{R}$ together with an element $\alpha \in M_x$ (which transports the uncertainty on $\mathbb{R}$ into elements of X). Two QBS measures (α_1, μ_1) and (α_2, μ_2) are equal if they define the same integrator: for all $f \in \mathbf{QBS}(X, \mathbb{R}_+)$,

$$\int_{\mathbb{R}} f(\alpha_1(r))\mu_1(dr) = \int_{\mathbb{R}} f(\alpha_2(r))\mu_2(dr)$$

In the following $\alpha_*\mu$ denote the equivalence class of pairs that defines the same integrator. We borrow the pushforward measure, as α should be thought of as a measurable functions transferring the measure on $\mathbb{R}$ into a measure on X. Then we can use the following notation for the integrator:

$$\int_X f(x)\alpha_*\mu(dx) \triangleq \int_r f(\alpha(r))\mu(dr)$$

Notice that fora any $x \in X$, the integrator of the constant random elements equal to x together with any real measure is independent of the measure. We denote this QBS measure $\delta_x \in \mathrm{Dist}(X)$, and we get:

$$\int_X f(y)\delta_x(dy) = f(x)$$

In the measure semantics, closed programs of type Tt are interpreted in $\mathrm{Dist}(t)$, the set of QBS measures in $[\![t]\!]$. We denote as $\rho \in \mathrm{Dist}(t)$ QBS measures, *i.e.* $\rho = \alpha_*\mu$ where $\alpha \in M_{[\![t]\!]}$ and $\mu \in \mathrm{Dist}(\mathbb{R})$

17. Show that for any QBS X, $\mathrm{Dist}(X) = \{\alpha_*\mu \mid \alpha \in M_X, \mu \in \mathrm{Dist}(\mathbb{R})\}$ satisfies the two first properties of QBSes[4] when endowed with

$$M_{\mathrm{Dist}(X)} = \{\beta : \mathbb{R} \to \mathrm{Dist}(X) \mid \exists \alpha \in M_X,\ \exists k : \mathbb{R} \to \mathrm{Dist}(\mathbb{R}) \text{ measurable },\ \forall r \in \mathbb{R}, \beta(r) = \alpha_* k(r)\}$$

3. It satisfies also the third one but the prove is more involved.
4. Of course it also satisfies the third QBS property.

We now adapt the semantics of terms in distributions to **QBS**:
Denoting $\texttt{Uniform}[0,1]$ as the usual uniform measure on $[0,1]$, and using the integrator notation and the δ_x notation introduced in the preceding lines, we define:

$$\begin{aligned}
[\![G \vdash \texttt{Dirac}\, e]\!]^{\mathrm{M}}_\gamma &= \delta_{[\![e]\!]_\gamma} \\
[\![G \vdash \texttt{unif}]\!]^{\mathrm{M}}_\gamma &= (\mathrm{id}_{\mathbb{R}})_* \texttt{Uniform}[0,1]] \\
[\![G \vdash \texttt{score}\, p]\!]^{\mathrm{M}}_\gamma &= [\![p]\!]_\gamma \,.\delta_{()} \\
[\![G \vdash \texttt{let}\, x = \texttt{sample}\; e_1\; \texttt{in}\, e_2]\!]^{\mathrm{M}}_\gamma &= \int_{[\![t_1]\!]} [\![e_2]\!]^{\mathrm{D}}_\gamma \,(x)\; [\![e_1]\!]^{\mathrm{D}}_\gamma \,(dx)
\end{aligned}$$

18. Give the interpretation of the `Lreg` program in **QBS**.

B.3. Importance sampling semantics

Consider the importance sampling semantics where closed programs of type $T(t)$ are interpreted as infinite trees whose leaves are labeled with weighted values of type t.

$$\mathrm{ImpS}(t) = \texttt{Tree}_\infty([\![t]\!] \times \mathbb{R}^+) = \left\{\texttt{Leaf}([\![\texttt{t}]\!] \times \mathbb{R}^+) \mid \texttt{Node}_\infty([0,1] \to \mathrm{ImpS}(\texttt{t}))\right\}$$

Each node represents the probabilistic choice made when sampling `unif` and each leaf contains the final value and the total weight of the trace of samples for the branch. This way, interpreting the program directly in $\mathrm{ImpS}(t)$ and sampling from the resulting tree corresponds to sample from the distribution infered by importance sampling.

19. Show that if $[\![t]\!]$ is a **QBS** then $\mathrm{ImpS}(t)$ is also a **QBS**.
20. Define the interpretation of the following terms:
    - $[\![G \vdash \texttt{unif}]\!]^{\mathrm{I}}_\gamma \in \mathrm{ImpS}(\mathbb{R})$
    - $[\![G \vdash \texttt{score}\, p]\!]^{\mathrm{I}}_\gamma \in \mathrm{ImpS}(\texttt{1})$
    - $[\![G \vdash \texttt{let}\, x = \texttt{sample}\; e\; \texttt{in}\, e']\!]^{\mathrm{I}}_\gamma \in \mathrm{ImpS}(t')$ (for $[\![G \vdash e]\!]^{\mathrm{I}}_\gamma \in \mathrm{ImpS}(t)$, $[\![G \vdash e']\!]^{\mathrm{I}}_\gamma \in t \to \mathrm{ImpS}(t')$)
    - $[\![G \vdash \texttt{Dirac}\, e]\!]^{\mathrm{I}}_\gamma \in \mathrm{ImpS}(t)$ (for $[\![G \vdash e]\!]^{\mathrm{I}}_\gamma \in [\![t]\!]$)
21. Compute the Importance sampling semantics of the `Lreg` program defined above.

In order to prove the corretness of the importance sampling semantics, we will refer to the measure semantics: it is sufficient to introduce a function $m^I : \mathrm{ImpS}(t) \to \mathrm{Dist}(t)$ which is compatible with the probabilistic constructions (sampling, dirac, unif and score) to prove that the importance sampling semantics is sound.

22. Define a QBS morphism $m^I : \mathrm{ImpS}(t) \to \mathrm{Dist}(t)$
23. Prove that:
    - $m^I([\![\texttt{unif}]\!]^{\mathrm{I}}) = [\![\texttt{unif}]\!]^{\mathrm{D}}$
    - $m^I([\![\texttt{score}]\!]^{\mathrm{I}}\; p) = [\![\texttt{score}\, p]\!]^{\mathrm{D}}$
    - $m^I([\![\texttt{Dirac}\, e]\!]^{\mathrm{I}}) = [\![\texttt{Dirac}\, e]\!]^{\mathrm{D}}$
    - $m^I([\![\texttt{let}\, x \;=\; \texttt{sample}\, e\, \texttt{in}\, e')]\!]^{\mathrm{I}}) = [\![\texttt{let}\, x \;=\; \texttt{sample}\, e\, \texttt{in}\, e']\!]^{\mathrm{D}}$

We have seen in the lectures that thanks to the law of large numbers, a large number of samples is needed to give a correct approximation of the distribution semantics.
The goal to approximate Bayesian inference is to find another representation for the program. It is possible to describe a representation accounting for the particle filter introducing the population sampling semantics where closed programs of type Tt are interpreted as

$$\mathrm{PopS}(t) = \texttt{Tree}_\infty(\texttt{List}(\mathbb{R}^+ \times [\![t]\!])) = \left\{\texttt{Leaf}(\texttt{List}(\mathbb{R}^+ \times [\![t]\!])) \mid \texttt{Node}_\infty([0,1] \to \mathrm{PopS}(t))\right\}$$

24. Describe the particle filter algorithm, and explain how you could prove its soundness taking inspiration in the demonstrated approach.